Отчет о создании “Онлайн обновление ленты новостей”.

Время создания: *02.08.2021*

1. Цель теста

Бизнес-цель теста – *получение навыков работы работа с WebSocket и очередями а также проектирование масштабируемых архитектур.*

Техническая цель теста – подключение к проекту резидентную система управления базами данных *Redis* с использованием кэшей и брокера сообщений.

2. Выводы

1. Для реализации масштабируемой архитектуры социальной сети с функционалом ленты новостей, была применена событи́йно-ориенти́рованная парадигма. В качестве системы обмена сообщений для бэкенд подсистемы используется сервис очередей сообщений *Redis PubSub*. Для доставки сообщений из подсистемы бэкенд до UI *React* пользователя используется протокол *WebSocket* с кастомной системой сообщений.
2. В разделе 4 приведены схемы и функциональное описание основных случаев использования данной архитектуры.

3. Настройки, redis

1. настройки минимальные, только добавился пользователь:
   `user hl on +@all ~* >password`
   для аутентификации и авторизации на использование сервисов *Redis*.

4. Функциональное описание

1. График 1. Случай запроса ленты новостей:
   По событию начала просмотра ленты новостей, на *шаге 1* открывается *WebSocket* до *endpoint*-а обслуживающего ленту новостей и посылается запрос *fetch* который запрашивает последние 1000 сообщений от друзей текущего пользователя.
   Если актуальные данные есть в кеше, на *шаге 2* новости читаются из кеша и отдаются по *WebSocket*-у на UI *React* пользователя.
2. График 2. Случай публикации новости:
   По событию публикации новости, на *шаге 1* посылается *REST* запрос на публикацию. На *шаге 2* данные сохраняются в БД *MySQL*. На *шаге 3* данные добавляются в кэш. На *шаге 4* публикуется сообщение в очередь *Redis PubSub*. У каждого пользователя свой канал. Текущий пользователь читает каналы друзей.
3. График 3. Чтение сообщения о опубликованной новости:
   По приходу сообщения из очереди сообщений *Redis PubSub* о публикации новости - *шаг 1*, данные запрашиваются из кеша и сохраняются в списке новостей текущего пользователя - *шаг 2*. На *шаге 3* данные при необходимости читаются из БД *MySQL* и сохраняются/обновляются в кеше. На *шаге 4* данные добавляются в кэш. На *шаге 4* по *WebSocket*-у посылается запрос *push* на клиента. Клиент получив запрос *push* может запросить *fetch* для реализации случая на 1-м графике.
4. Графики

График 1. Случай запроса ленты новостей.

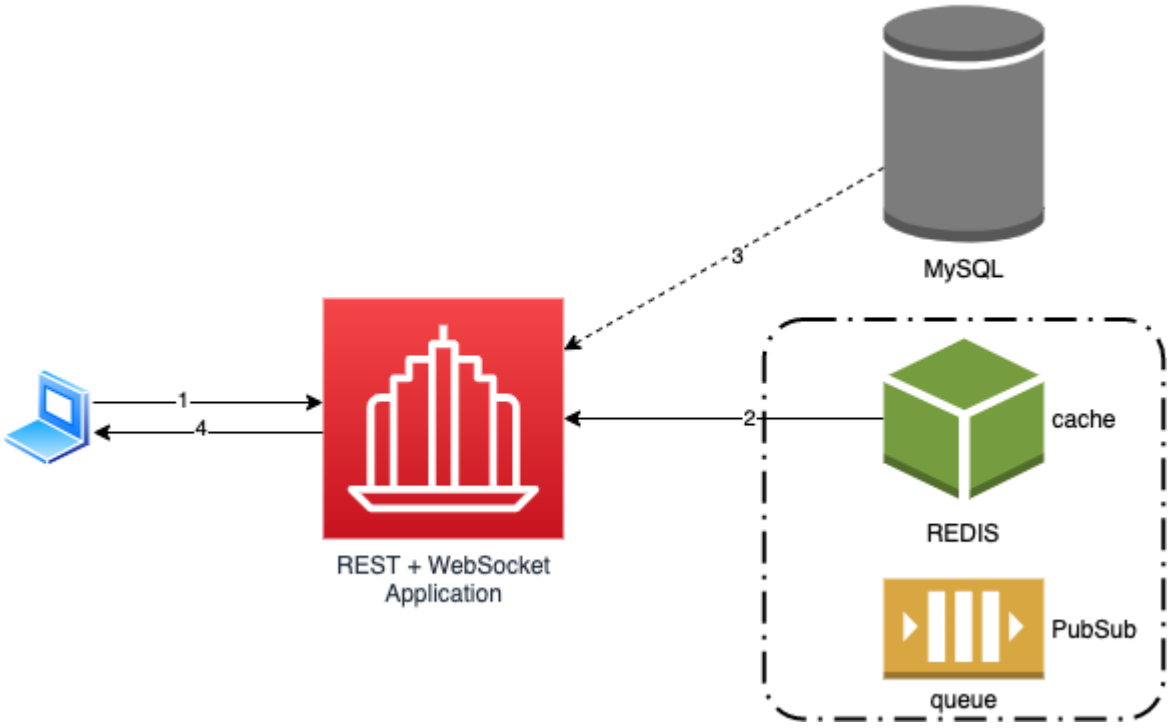

График 2. Случай публикации новости.

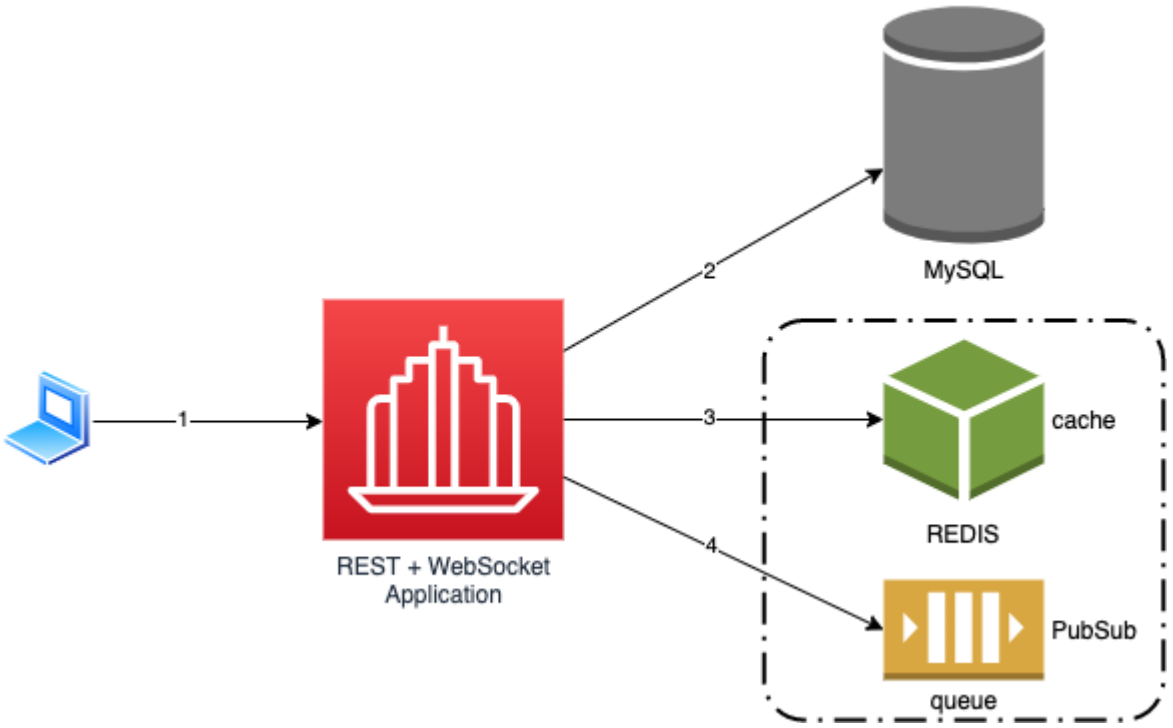

График 3. Чтение сообщения о опубликованной новости.